Thesis Report

Thesis Title: Soft Shadow Rendering in Real Time

Shihab Rezwan Manzur
ID: 09201028
CSE Department
BRAC University

Supervisor: Professor Dr. Mumit Khan
Date: 13 - 08 - 12

Abstract

Shadows have become an integral part in today's applications. Due to the advancement of recent GPU (Graphics Processing Unit) systems, shadows can be implemented in real time. Shadows give us a total picture of the scene and make the applications more realistic. Applications, more likely computer games ask for shadows, to get a precise idea about characters position in the scene. Previously only hard shadows were present, cause of lacking of decent GPUs. Now there are better and more advanced GPU system are available. This makes it possible to design soft shadows in applications. Designing soft shadows asks for complicated algorithms and rendering shadows in real time is trickier. In our paper we kept it real simple to render soft shadows. We have done some basic geometric calculations to achieve the idea about the scene and drawn a hard shadow and then made it soft.

1. Introduction

Recent advanced GPU systems have made it evident to add shadows in movies, games and applications.

What is shadow? Why it is necessary?

Shadow is a dark area caused if a body comes between light source and surface area. Shadow is needed to bring realism in image. Also shadow helps to understand object placement in 3D scene.

Shadows are of two types

1. Hard Shadow
2. Soft shadow

Previously hard shadows were used because of hardware constraints. Hard shadows depend only on one binary notion and that is whether the point is in shadow or not. Hard shadows are relatively easier to compute but don't generate outstanding results. There is another problem, to compute hard shadows point light source is used, which doesn't exist in real life.

Soft shadows are introduced, which generate more realistic images to the viewer. Instead of a point light source an area light source is used. Object placement in 3D space is more accurate with soft shadows. Shadow is divided into umbra (dark region on the surface) and penumbra (relatively less darker region) and combining these soft shadows are generated.

Several techniques to render shadow are introduced, like

- Raytracing
- Radiosity
- Phong Shading
- Variance Soft Shadow Mapping

- Convolution Shadow Mapping
- Percentage Closer Shadow Mapping
- Layered Variance Shadow Maps
- Image Based Methods etc.

Not all this methods produce soft shadows in real time. Raytracing, radiosity render most realistic images with shadows but asks for a very complex computation, and result is not generated in real time. Thus we not look into those techniques. Due to the advancement in recent GPU systems soft shadow rendering in real time has become more common. Thus new techniques are being introduced.

Rendering soft shadows generated from extended light source in real time is an ongoing research topic. Till today no single method is been introduced to render shadows, thus researches are going on. We look onto develop our own real time shadow rendering method or modify existing ones.

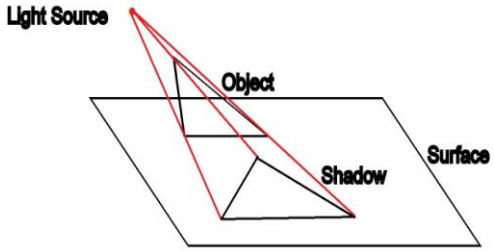

Figure 1: Basic Way to Draw Shadow

2. Some Basic Tools for shadow rendering

Shadow rendering is a delicate thing to do and there are a vast array of factors which play a part in drawing shadows and idea about these factors are the priors to draw shadows.

2.1. Light Sources

Light sources also play a vital role in rendering shadows. In CG(Computer Graphics) point light source is used to draw hard shadows. In real life point light sources don't exist. Hard shadows also give us a not so realistic view about the scene.

Then comes, determining shadows with multiple light sources. This can be achieved easily if we calculate drawing a shadow for single light source.

2.2. Umbra and Penumbra Region

In rendering shadows, there are a couple of complicated factors. One such is, knowing the umbra and penumbra area.

Umbra region is where the light source is totally occluded by the object and a hard shadow in drawn onto the surface by the size of that area. Penumbra is the area where the light source is not occluded and a softer shadow is drawn by the area. Combining these two a complete scene with soft shadows is accomplished.

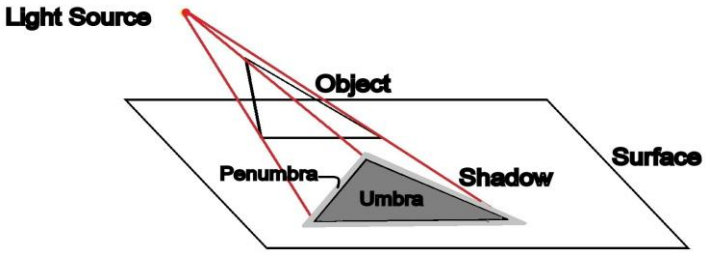

Figure 3: Umbra and Penumbra Region

2.3. Distance Factors

In shadows the distance between objects, lights source and surface also plays a vital role. Whenever a change in distance between the occluder and the light source occurs the shadow will change its shape and make a realistic view. If the distance between the light source and object is high then a smaller shadow regions should be drawn, if the distance between the lights source and the object is low there should be a larger shadow region.

3. Shadows in Real Time

We are into developing an application which renders soft shadows in real time. We won't be looking at the techniques which ask for high calculation time also we will be overlooking the shadow quality.

Nowadays high end GPU systems are available which calculates shadows fast and thus determining real time requires some calculations. We will be looking at the FPS(frames per second) to get the actual idea about rendering time.

4. Survey of Shadow Algorithms

The first complete survey on soft shadow algorithms were done by Woo et al [90]. According to them complexity for rendering shadows depends on three basic things, which are Storage Usage, Pre-processing Runtime complexity and complexity during the actual rendering is done.

Hard shadows are easy to compute. Only one thing is to be kept in mind while rendering hard shadow and that is to know whether the points are in shadow region or not.

To render soft shadows there are a couple of techniques are discussed. Frame buffer algorithm, distributed ray tracing, cone tracing, area subdivision, radiosity are some of the techniques used to render soft shadows, though all of them are not into rendering shadows in real time.

GPU systems have been highly developed and now focus onto real time rendering is increased. Image quality is also improved as antialiasing, motion blur, shadow casting are become general.

There are several things to be kept in mind whenever it comes to rendering shadows, like position and size of the occluder, geometry of the occluder, geometry of the receiver etc.
Recent growth in computer graphics technology has made it a reality to render 3D graphics in real time. Thus, previous rendering techniques are to be updated for better results.
Hasenfratz et al [03] has done a survey on shadow algorithms in real time. We are to study these algorithms.

In general shadows are of two types, hard shadows and soft shadows. Hard shadows are comparatively easier to compute. In hard shadows only one thing needs to be in focus, whether the point is in shadow or not. Hard shadows are generated by point light source. In real life, point light source doesn't exist. There are algorithms to implement hard shadows.

Soft shadows are more complicated to implement. It gives more realistic view than hard shadows but requires high complexity.

Softness of shadows depends on the distance between the source, occluder and receiver. It is not easy to compute soft shadows. It calls for complex techniques. There are some issues to be dealt, when it comes to render soft shadows. Practically single light source doesn't exist. Shadows produced by several light sources are to be calculated. This can be handled easily if shadows from single light source are obtained. Then added together and final shadow is achieved.

There might be more than one occluder. To get the shadow, the shadow area for each occluder is added to get the final shadow region. It is not easy to calculate the relation between partial visibility functions of different occluder, thus approximate values are used.

Computing shadows for extended light source is complex to compute. In real time, shadow algorithms compute visibility information for one point and then using visibility information behavior for the extended light source is calculated. Using complete extended light source for visibility computation is algorithmically too complex and can't be used in real time. Dividing the light source into smaller light sources removes the complexity to some extent.
Penumbra region generation is a process to generate soft shadows from hard and done in real-time. Algorithms are there to compute penumbra region, as

1. Extend the umbra region outwards, computing outer penumbra region.
2. Shrink the umbra region inwards, compute inner penumbra region.

We focus onto render soft shadows in real time. So, we don't look into techniques like ray tracing, radiosity, Monte-Carlo ray tracing or photon mip mapping as these are complex to implement and don't render shadows in real time.

Shadows from point light source, shadow mapping and shadow volume algorithm will be the point of focus.

To compute shadows one thing is a must, that is to identify the parts of the scene which are hidden from the light source; this is same as visible surface determination.

There comes the shadow map, Z-buffer.

At first computing the view of the scene is done, from the point of view of the light source. Then Z values are achieved and kept in the Z buffer. Z values hold the geometric position of each pixel, if changes do not occur the values are same or changed accordingly.

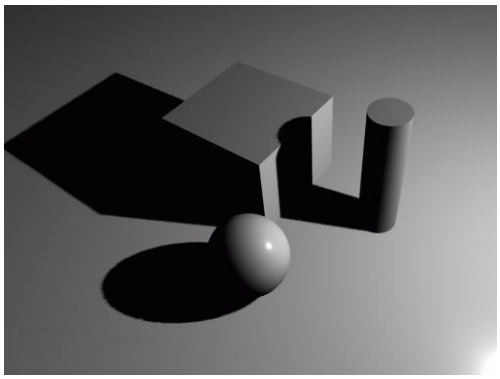

Figure 2(a): Hard Shadow

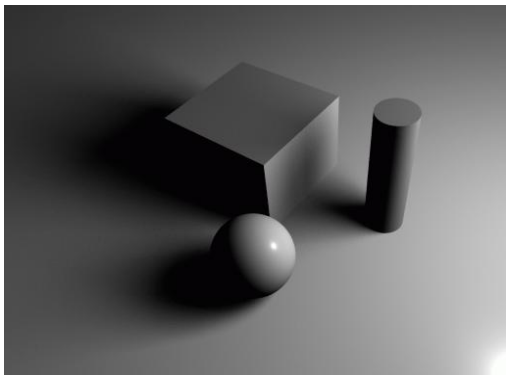

Figure 2(b): Soft Shadow

5. Some Techniques to draw soft Shadows

Hasenfratz et al [03] has done a survey on shadow algorithms in real time. We are to study these algorithms.

In general shadows are of two types, hard shadows and soft shadows. Hard shadows are comparatively easier to compute. In hard shadows only one thing needs to be in focus, whether the point is in shadow or not. Hard shadows are generated by point light source. In real life, point light source doesn't exist. There are algorithms to implement hard shadows.

Soft shadows are more complicated to implement. It gives more realistic view than hard shadows but requires high complexity.

Softness of shadows depends on the distance between the source, occluder and receiver. It is not easy to compute soft shadows. It calls for complex techniques. There are some issues to be dealt, when it comes to render soft shadows. Practically single light source doesn't exist. Shadows produced by several light sources are to be calculated. This can be handled easily if shadows from single light source are obtained. Then added together and final shadow is achieved.

There might be more than one occluder. To get the shadow, the shadow area for each occluder is added to get the final shadow region. It is not easy to calculate the relation between partial visibility functions of different occluder, thus approximate values are used.

Computing shadows for extended light source is complex to compute. In real time, shadow algorithms compute visibility information for one point and then using visibility information behavior for the extended light source is calculated. Using complete extended light source for visibility computation is algorithmically too complex and can't be used in real time. Dividing the light source into smaller light sources removes the complexity to some extent.
Penumbra region generation is a process to generate soft shadows from hard and done in real-time. Algorithms are there to compute penumbra region, as

3. Extend the umbra region outwards, computing outer penumbra region.
4. Shrink the umbra region inwards, compute inner penumbra region.

We focus onto render soft shadows in real time. So, we don't look into techniques like ray tracing, radiosity, Monte-Carlo ray tracing or photon mip mapping as these are complex to implement and don't render shadows in real time.

Shadows from point light source, shadow mapping and shadow volume algorithm will be the point of focus.

5.1. Shadow mapping

To compute shadows one thing is a must, that is to identify the parts of the scene which are hidden from the light source; this is same as visible surface determination.

There comes the shadow map, Z-buffer.

At first computing the view of the scene is done, from the point of view of the light source. Then Z values are achieved and kept in the Z buffer. Z values hold the geometric position of each pixel, if changes do not occur the values are same or changed accordingly.

Some techniques to render soft shadows in real time,

- Image based approach
- Object based approach

5.2. Image based approaches

- Combine several shadow textures taken from point samples on the extended light source
- Using layered attenuation map, instead of depth images
- Using a couple of numbers shadows maps
- Convolving a shadow map with an image of the light source

5.3. Object based approaches

- Combine several shadow volumes taken from point samples on the light source
- Shadow volume is broaden
- Penumbra volume for each edge of the shadow silhouette
- After all is done it must be put into mind that rendering has to be done in real-time.

Geometry of the occluder, physically exact shadow, methods computing only inner or outer penumbra should use approximate value, light source, number of subdivision of light source, drawing the silhouette of objects influences rendering shadows in real time.

Image based techniques generate high quality soft shadows. This technique also works with area light source. The complexity of image based process depends upon the image size and light source rather than on scene complexity. This technique does not rely on the geometry, so result is achieved faster.

In the paper two techniques are discussed

- Layered attenuation map
- Coherence based raytracing of depth images

Soft shadows from area light source bring realism on images generated by computers. Determining the penumbra still remains a problem to be dealt with. To determine the penumbra computing visibility between every surface point and every light source is needed.

William [78] did some works on computing hard shadows from point source. He has done visibility calculation in image space. This method does not give good results due to under sampling and aliasing artifacts. Also point light source doesn't exist in real life.

Agarwala et al [2000] has generated soft shadows merging two techniques. Both of the techniques compute shadows in image space. Time and memory for this technique depend on image size and number of lights. Thus, geometric complexity is removed.

5.4. Layered Attenuation Map

This approach has interactive rendering rates, but sampling flexibility is limited. (LDI)Layered depth image is computed first. The LDI keeps track of the depth information and layered attenuation maps. This helps to generate projective soft shadow textures. Proper attenuation is used for normal rendering but shadows.

5.5. Coherence Based Raytracing

This approach is good to generated high quality images. Not an interactive method. Shadow maps are computed first from few points of the light source, often boundary vertices. Visible portion of the light source does not very for surface point close to one another.

Object based methods like distributed raytracing, radiosity, discontinuity meshing or back projection generate soft shadows. These techniques provide good result regarding shadows but are not fast.

Soler and Sillion [98] used convolution on blocker images for generating approximate soft shadows. This technique avoids sampling artifacts but cluster geometry in object space is there. Clusters cannot shadow themselves. Plants or trees require a large number of clusters, thus complexity is increased.

Layered attenuation map method give good results when it comes to rapid previewing, has fast pre-computation phase and interactive display phase, independent of scene geometry complexity. Final images for applications such as pre-rendered animation ask for anti-aliased, artifacts free shadows. To do so coherence based ray-tracing algorithm is introduced.

Soft shadow mapping has become a very captivating part regarding computer applications and games. Shadow mapping has to be very efficient if to be done in real time. Previously image based shadow mapping were used to map shadows. This technique is used to map hard shadows. To map soft shadows many method have been introduced. VSM (Variance Soft Shadow Mapping), PCSS (Percentage Closer Soft Shadows) are a few of the methods to map soft shadows. When the light source is fairly small this methods afford to generate better results, but with the extension of light source this methods don't perform correctly. Yang et al[2010] has tried to improve PCSS method to achieve enhanced performance on the subject of mapping soft shadows. Variance Soft Shadow mapping is a widely used method for mapping soft shadows requires pre-filtering and also needs a lesser amount of memory space to keep textures. But VSM doesn't always map soft shadows with accuracy. It might generate error while computing average blocker depth values; in consequence brute force is applied. VSM method is also not so superior while dealing with soft shadows. Some pixels may be inaccurately lit leading soft shadow not to be proper. This problem is called non-planarity problem.

Yang et al [2010] aimed to design a formula for estimating average blocker depth, based on VSM theory. They also intend to develop practical filter kernel subdivision scheme to handle non-planarity problem. There are two types of subdivision, uniform way and adaptive way. Either can be chosen for implementation. Existing shadow algorithms are used to map shadows, henceforth Yang et al [2010] has done so. Previously classical discontinuity messing was used to generate shadows. in recent works researcher prefer shadow mapping techniques. There is a method named back-projection which is very complex to be implemented in real time and also yields incorrect occlusion or light leaking. Hard shadow mapping has some problems of its own. Pre-filtering and edge anti-aliasing are some of those. Pre-filtering cannot be executed with a

proper shadow test. Some pre-filtering methods are implemented to run shadow tests. VSM supports pre-filtering very well. There is another method named ESM (Exponential Shadow Maps), which uses exponential functions to run shadow test operations. These functions are dependent on PCF (Percentage Closer Filtering) for non-planarity, which works fine with soft shadows. But when mapping soft shadows, PCF is very expensive process. Soft shadow mapping with pre-filtering might show non-planarity problems.

Variance soft shadow mapping is based on Chebyshev's inequality and results in close approximation, not accurate always. Most talked about problem in PCSS and VSM is how to calculate average blocker depth with efficiency? There is also the non-planarity problem. Yang et al [2010] has proposed some ideas to solve these problems. They divided the main kernel into subdivision kernels of equal size. Then they chose PCF sampling over assumption based lit kernel. PCF removes non-planarity problem to some extent if the sampling size is small. Two types of subdivision are used in shadow mapping, uniform and adaptive. If the number of sub-kernel is large (>= 64), adaptive subdivision gives better results. Adaptive kernel might give a better result. Yang et al [2010] proposed to merge both of these subdivision schemes and achieved a better result. Yang et al [2010] has used SAT (Summed Area Tables) to pre-filter. But SAT has numerical precision loss for large filter kernel. Hence 32-bit integer is used. Z avg. and d is calculated and there difference is saved, then comparing with some given value, outcomes are accomplished.

Yang et al [2010] has used all the present technologies of mapping soft shadows and showed us the difference between all of those. They used ray tracing, VSSM, PCSS, back-projection to map soft shadows and different output is achieved and presented.

Shadow computing is a problem in graphics. With extended light source the problem grows. Hence convolution technique is introduced which computes shadows fast and accurately also.

Soft shadows are achieved through variation of illumination on surface. If the light source is occluded by other objects, then visibility is determined by penumbra region.

The calculation required to render soft shadows is complicated to implement. Then visibility determination should also be handled. Many methods have been introduced to render hard shadows from point light source. Computing shadows from area light source is still a problem in computer graphics.

Soler and Sillion[1998] has done soft shadow rendering with artifacts free image in an efficient way.

They calculated shadow map. Textures are created from images with light source and occluders. Then these textures are used to render soft shadows. The convolution method is run offscreen buffers. Exact images are needed for parallel objects otherwise approximation is used. Overall approximation is handled by clusters. Adding shadow maps from sub-clusters are added and result is achieved.

Single rendered image is used, so shadows are rendered fast and interactive rate can be accomplished.

All shadow algorithms do not work in real time. Ray tracing algorithm cast shadows from a point in the surface towards light source. Ray tracing creates realistic images with shadows, but an expensive process.

Depth images are used to store information about the image with respect to light source and later on shadows are added based on the information. This process cannot handle extended light source, thus penumbra region might be erroneous. Aliasing problem is not handled in a proper way.

Solar and Sillion has stated about using a different data structure to handle the information required to render shadows. Shadow volumes can be used, using point light source and occluders. Point light source is not very useful when it comes to render soft shadows.

To keep record about the visibility information due to extended light source discontinuity mesh can be used. Both the structures are complicated for computation.

Shadow generating itself is a complicated process and to make it interactive extra cautions are needed to be taken. Till the research of Woo et al [90] no single method has been put forward regarding shadow rendering.

Shadow construction can be done in the back while the scene is rendered. This requires hardware acceleration. Due to recent advanced GPU systems this can be done.

Interactive pre-calculating of soft shadows were first done by Heckbert and Herf[98]. They created a number of shadow images for points on the light source and combined later. This process asks for high end graphics cards and hard shadows are achieved instead of soft.

6. Our Technique

We are focusing on real time soft shadow rendering. Avoiding all the techniques which needs high calculation, more memory size and high end GPU systems were our main concern. We have implemented a very simple method to draw soft shadows. We used simple geometry to know about the scene and drawn objects.

We have drawn a triangle object and a surface. Then we calculated the normal for the surfaces using basic geometry.

When two lines are perpendicular to each other, then they are called normal. The surface normal, a tangent is drawn over the curved surface and the line or vector perpendicular to the tangent plane is called the surface normal.

For a plane given by the equation $ax + by + cz + d = 0$, the vector (a, b, c) is a normal.

For a plane given by the equation, $r(\alpha, \beta) = a + \alpha b + \beta c$.

Doing so we get the surface position were our shadow would be drawn.

We then use matrix multiplication to get the actual points for drawing the shadow onto the surface. Matrix multiplication takes two matrices and multiplies their elements and gives us a new matrix with the resultants. The size of the two matrices has to be same, or else multiplication will not take place. The resultant matrix is also the same size as the two matrices.

We also use another basic geometry equation. Equation of a straight line is,

$y = mx + c$. at first we calculate the tangent line between all the vertex point of the object and light source. Then using $y = mx + c$ equation we derive the project points of the shadow onto the surface.

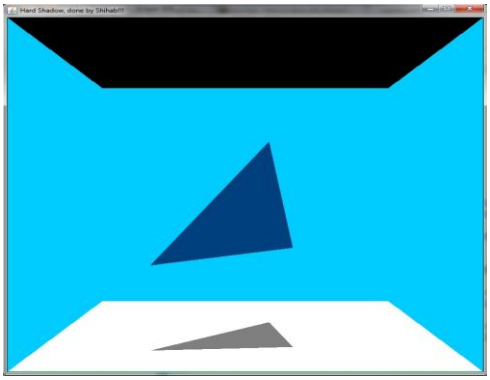

Figure 3(a): Hard Shadow

Figure 3(a) shows a hard shadow drawn using our method.

To draw the soft shadow, we just draw the hard shadow first and then we made it softer.

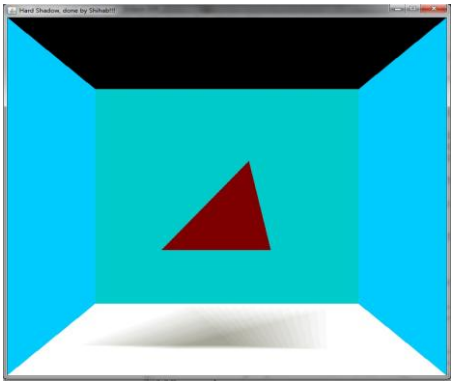

Figure 3(b): Soft Shadow

7. Advantages

From the very beginning it is stated that we are to design soft shadow in real time. Our algorithm is very basic and it is at liberty of other high calculation based techniques.

New advanced GPU systems come with a lot of functionality. These GPUs support all the basic functionalities regarding rendering shadow with different techniques. Programmable GPUs have also emerged and supports single sample soft shadows, penumbra maps, smoothies and soft shadow volumes.

We don't need to calculate the silhouette of the object. Calculating the silhouette asks for high pre-computation. Silhouette calculation cannot be done in GPUs and done in main processing unit. So, pre computation is quiet lengthy.

We also don't need a very high memory space. As we don't need to keep track of the Z values and also other buffers like stencil buffer is not needed. So, large memory space is not required.

8. Results

We draw the tables with fps values for hard shadows and soft shadows.

Time(sec)	FPS
1	18
2	24
3	23
4	14
5	15
6	26
7	24
8	13
9	15

Figure 4(a): FPS for hard shadow

Time(sec)	FPS
1	21
2	24
3	22
4	22
5	20
6	23
7	21
8	19
9	21

Figure 4(b): FPS for soft shadow

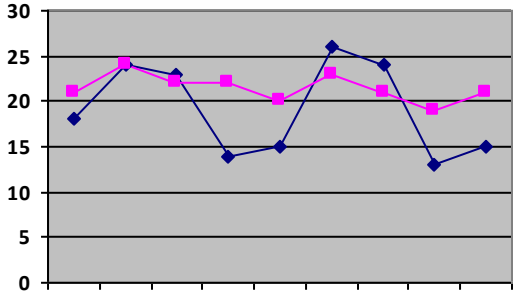

Figure 4(c): FPS Comparison

9. Conclusion

We were able to render our soft shadow with a lower GPU system and we had a better result than expected. We discussed about some existing real time soft shadow algorithms. We designed a very simple but efficient soft shadow algorithm based on geometry.

10. Hardware

We have implemented our soft shadow rendering algorithm in a system with Intel Core 2 Duo CPU, E7300 @ 2.66GHz, Ram 4.0GB, Graphics card NVIDIA GeForce 9500 GT, 1GB.

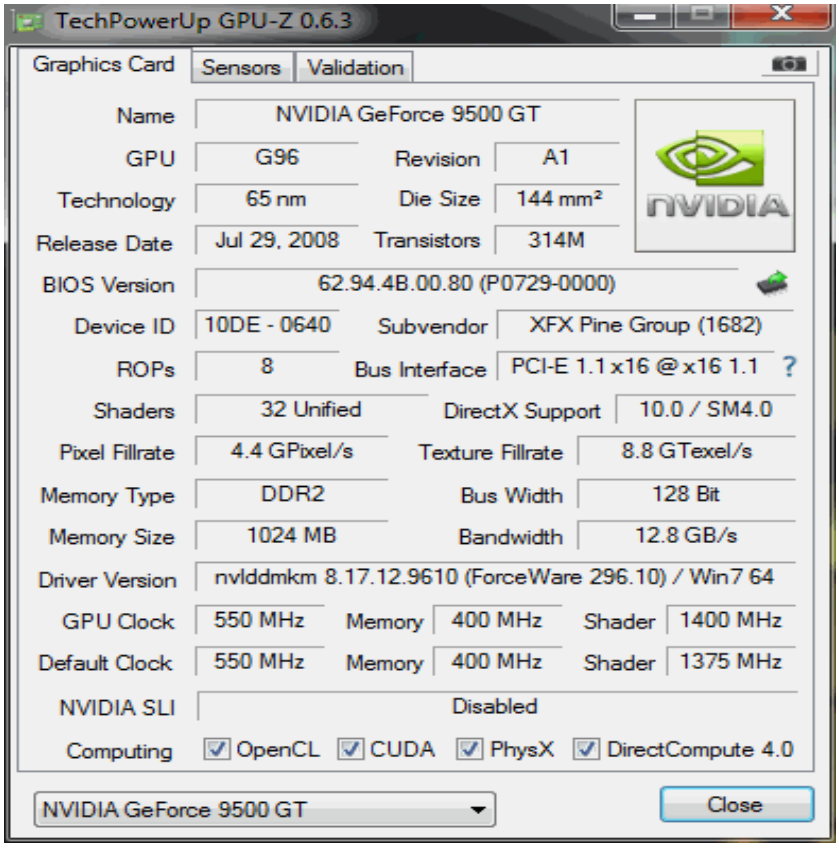

Figure 5(a): Information about hardware

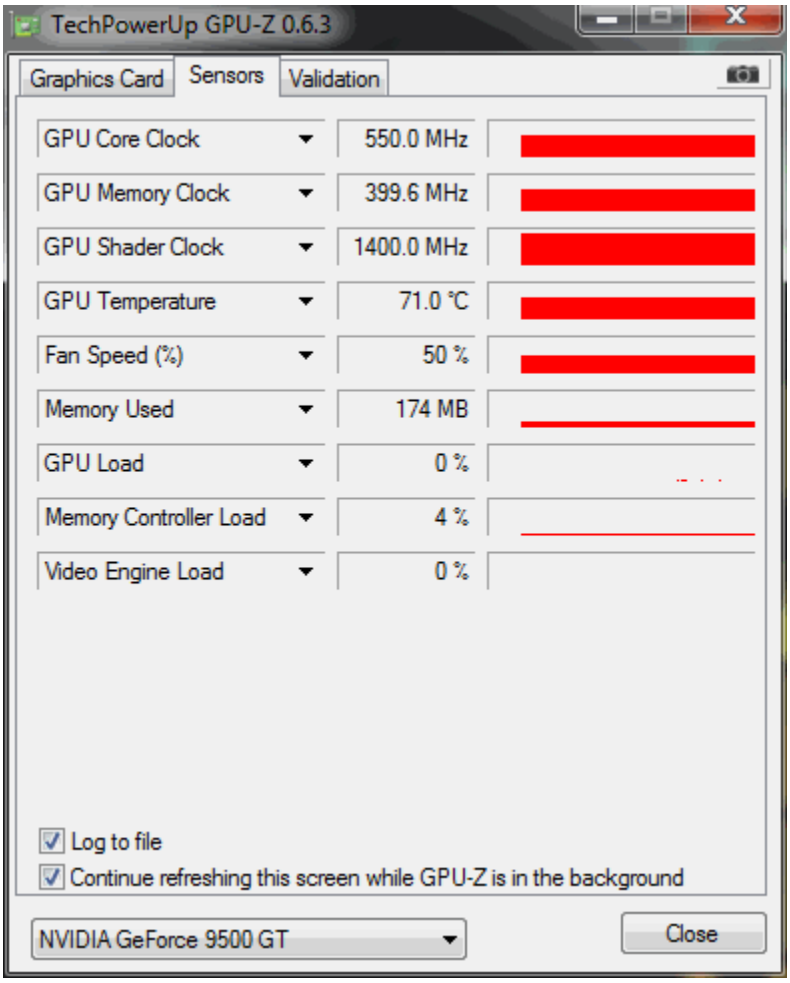

Figure 5(b): Information about hardware's state

11. Software and Languages

While implementing soft shadow in real time we used Eclipse Indigo IDE, jogl 2.0 pack for Eclipse, gluegen 2.0 pack for Eclipse. We calculated fps for hard shadows and soft shadows separately. For this we used FRAPS software. We also used GPU-Z o know about hardware state more precisely.

12. Future Work

Our algorithm works with low end graphics hardware and lacks quality. With the functionality that comes with recent advanced GPU systems we can achieve better results. Our design works with a very simple scene. We look forward to develop as soft shadow algorithm in more complex scene and time will not exceed to a very high range. We will try to use our algorithm for rendering soft shadows with multiple lights source and multiple objects also.

13. References

1. A. Woo, P. Poulin, and A. Fournier, .A Survey of Shadow Algorithms., *IEEE Computer Graphics and Applications,* **10**(6):13.32, November 1990.
2. L. Williams, .Casting Curved Shadows on Curved Surfaces., *Computer Graphics (Proceedings of ACM SIGGRAPH 92)*, pp. 270.274, August 1978.
3. AGRAWALA, M., RAMAMOORTHI, R., HEIRICH, A., AND MOLL, L. 2000. Efficient Image-Based Methods for Rendering Soft Shadows. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press/ACM SIGGRAPH, New York. K. Akeley, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 375.384.
4. J.-M. Hasenfratz, M. Lapierre, N. Holzschuch, and F. Sillion, "A survey of realtime soft shadows algorithms," Computer Graphics Forum, vol. 22, no. 4, pp. 753–774, 2003.
5. Knill, D.C., Mamassian, P., Kersten D., "Geometry of shadows," J. Opt. Soc. Am. A/ Vol. 14, No. 12/ December 1997
6. William Reeves, David Salesin, and Robert Cook, "Rendering Antialiased Shadows with Depth Maps," *SIGGRAPH Proceedings '87*, 21(4), pp. 283-291, 1987.
7. Heidmann, T. 1991. Real Shadows, Real Time. Iris Universe, 18(November), 23-31
8. Assarsson, U., Moller, T.A., "A Geometry-based Soft Shadow Volume Algorithm using Graphics Hardware", Chalmers University of Technology, Sweden